## 1. PostgreSQL JOIN Performance

I used the Neo4J flights data to create the flights database in PostgreSQL including the following tables: flights, airlines, airports. This simple data model was used in this performance comparison. Given that the data contained less than 30 rows, this performance test does not represent a real-world scenario.

The following SQL SELECT statement was composed for the performance comparison. The tables are FOREIGN KEY’d on their PRIMARY KEY’d id columns, but on index is present on the flights table’s columns.

|  |
| --- |
| SELECT a.name AS AirlineName, d.city AS Departed, o.city AS Arrived, f.\* FROM flights f  INNER JOIN airlines a ON a.id = f.airlineid  INNER JOIN airports d ON d.id = f.depart\_airportid  INNER JOIN airports o ON o.id = f.arrive\_airportid; |

|  |  |
| --- | --- |
| # | Execution Time |
| 1 | 32 ms |
| 2 | 32 ms |
| 3 | 31 ms |
| 4 | 31 ms |
| 5 | 31 ms |

Next, the following index was added to the flights.airlineid column:

|  |
| --- |
| CREATE INDEX ndx\_airlineid ON flights (airlineid ASC); |

Again the SQL SELECT statement from above was executed and execution time was measured.

|  |  |
| --- | --- |
| # | Execution Time |
| 1 | 16 ms |
| 2 | 16 ms |
| 3 | 31 ms |
| 4 | 16 ms |
| 5 | 31 ms |

Next, the following index was added to the flights.depart.airportid column:

|  |
| --- |
| CREATE INDEX ndx\_depart\_airportid ON flights (airlineid ASC); |

Again the SQL SELECT statement from above was executed and execution time was measured.

|  |  |
| --- | --- |
| # | Execution Time |
| 1 | 32 ms |
| 2 | 31 ms |
| 3 | 31 ms |
| 4 | 15 ms |
| 5 | 31 ms |

Next, the following index was added to the flights.arrive.airportid column:

|  |
| --- |
| CREATE INDEX ndx\_arrive\_airportid ON flights (airlineid ASC); |

Again the SQL SELECT statement from above was executed and execution time was measured.

|  |  |
| --- | --- |
| # | Execution Time |
| 1 | 31 |
| 2 | 32 |
| 3 | 31 |
| 4 | 32 |
| 5 | 15 |

What conclusions can be drawn from our execution time results? At best, PostgreSQL query execution benefits slightly with the index on this small amount of data but where indexes are really beneficial is on larger data sets. Given the small size of this data, it is easy to think they are not needed, but as more data is loaded, PostgreSQL will begin to rely more and more on them.

## 2. Advanced Create Table

## 3. R Performance

A study in performance of summation between the standard for loop, base R’s sum function and a parallelized summation using doParallel and parallel packages was performed. The final paper was uploaded to Rpubs and is available at <http://rpubs.com/dwdii/R-SumInParallel>.

All code associated with this Quiz is available in [my DataAcqMgmt GitHub](https://github.com/dwdii/DataAcqMgmt/tree/master/Week13) repository.